$2.00

VOLUME 2 · 1987
VER 21
MARCH

R/D COMPUTING

Dedicated to TI 99/4A and 9900 Computer Systems

.....Ryte Data.....

210 MOUNTAIN STREET,
HALIBURTON, ONTARIO K0M 1S0
(705) 457-2774

Greeting and salutations. Nothing is sacred. Consider "Information As A Weapon". Particularly in an industry that moves at an astounding pace. Whatever is KNOWN is obsolete. Whatever in NOT known (by them) is a competitive tool. If it doesn't have bugs, it isn't new enough.

Beyond the battle ground, out towards the cutting edge are individuals who dream up the future. Here we have several visionaries who would like to communicate with others regarding common ideas and projects.

Robert Wagner, 114-2 Sherwood Circle, Minot AFB, ND 58704 would like to co-ordinate a hardware group. This activity could easily design various interface units and innovative hardware projects. Mr. Wagner discussed the advantages of having schematics, specs, drawings and finished boards available for many of the projects within the TI 99/4A community.

We have found the expense and cost of having CAD electronic drawings and printed circuit boards made REQUIRES a large audience.
Wire-wrap prototypes are so tedious. Therefore, we would like to make a call to all hardware / technically inclined TI owners to contact Mr. Wagner at the above address. This could develop into a major hardware forum for the 99/4A and the Geneve.

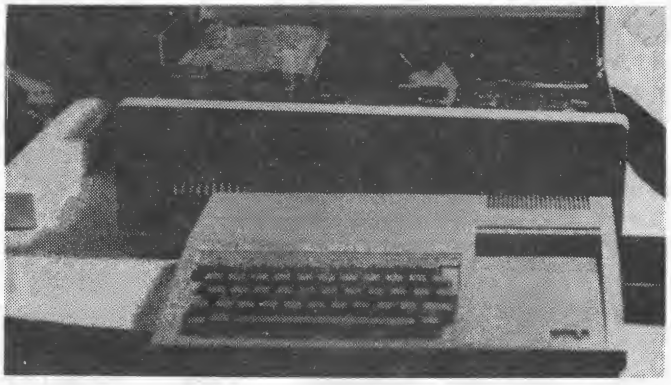
We will offer R/D COMPUTING as a tool to communicate and diseminate information to a wider audience and between those who participate.

A number of ideas regarding possible future projects: a digitizer device, a hardcard 20 meg drive on a card, an extended memory specification, optical disk interface, videodisk controller, optional video display input device etc. were discussed. Write Robert today with your views & ideas! Send us a copy. Do it NOW!

THE BOX... AGAIN.

Third time is a charm. What you see here is the 99AT Expansion System.

The expansion card design, mechanical layout, computer assisted final design artwork, numerical control tape, negatives, board etching, tooling and finishing are all completed. Features: Five standard expansion card slots, 130 watt power supply with power supply connectors, room for four half-height disk drives, cooling fan and self contained interface card. All five slots are useable.



Options include: First option is to install a flex cable to connect a standard TI 99/4A computer. Second option involves the use of a Myarc GENEVE computer inside the expansion box. Third option involves installing the main TI 99/4A

motherboard inside the case. An external keyboard and GRAM Card would complete the system. Fourth option, applicable to all systems, includes a larger power supply. Standard model contains a 130 watt power supply. Both 150 watt and 200 watt supplies can be installed without futher modification.

Pictured here is the 99AT Expansion System with a RAVE Keyboard and GRAM Card installed. All ports have been brought out to locations on the expansion system.

The first option requires a flex cable to be installed on the interface. We have a suitable shielded cable available to be installed prior to shipping. This cable is round and attaches to the side I/O port with the cable at the back. This cable can also be used to interface a 99/4A console at a later date by simply plugging it in.
The second option requires no other modifications. Installation of a Myarc 9640 involves plugging the card into the bus.
The third option does require modification of your case to accept the necessary mounting hardware and power supply connectors.

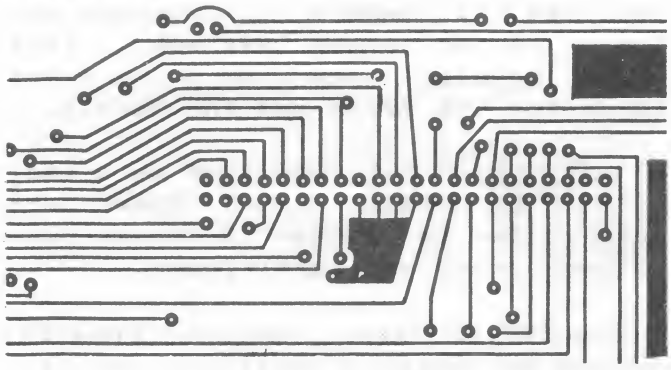
Orders are being accepted. Based on a guaranteed shipping date - according to the sequence orders are received. A short waiting list has evolved from this development.
You must specify the options you require at the time you place your order ie: use for a 4A or Geneve.

Due to the unique situation regarding the Myarc 9640, we are slated to take delivery of a full production model at the end of March. We will have a full report on the operation of a production model at that time.

INPUT: from Tony Lewis

The February issue of "Computer Design" had an article discussing single board computers. Two companies, one in Virginia and the other in California which sell 99xx based computers. The VA firm sell a STD bus single board computer based on the 9995 for $350.00. The company in CA sells a 99/05A (yes, the fabled 99 thousand series) computer system for $4000.00. More info to follow.

If I might be permitted to get back up on my soapbox and preach - the discovery of these 'other' TI computers ispired me once again to do a single board 9900/9995 computer for the PE box. I still believe there is a rabid bunch of TI hackers that would love to have a $150 kit (a la' Horizon RAMdisk) to put a 256k RAM computer on a card in their PEB, expandable to 2Mbytes. If anyone can release the hardware to the public, the various assembly people will write the application programs. For example, configured as a 'C' machine and interfaced to the 99/4A via the multi-µP chip. I think people want to SEE something, not just rumours or drawings.



One thing that keeps technology going is COMPETITION. If I do successfully design a single board computer, I hope two other people design boards twice as good. The more the merrier! Perhaps some tech articles will inspire people in the TI community to do it themselves rather than wait for Tony Lewis, Ryte Data, MYARC, CORCOMP etc. to do everything for them. Otherwise the 99xx family will cease production and disappear forever. Oneday I hope to be able to convert the single board computer to the IBM PC bus - thus extending the life of th microprocessor for people who jump ship to cheap clones.

Tony Lewis - Feb/87

Introducing: ACCESS ENGINEERING INC. (AEI).

Description: AEI was founded in the spring of 1986 specifically to produce products for the newly developed Myarc 9640 computer. The creation of this company and the associated risks is based on analysis that concluded that; both the technology and the market potential of the new computer is significant/ AEI brings to the Myarc 9640 marketplace not only the expertise of its founders and dealer network, but a development facility that already contains 6 microprocessor types along with a mini computer based development system. AEI has or is in the process of developing a series of professional level products to take advantage of the speed, memory and graphics of the Myarc 9640. We, after careful evaluation, believe to be a superior machine in several levels of the marketplace.

Our Founders:
JIM HORN: The SYSOP (SYStem OPerator) of the H & R Block firm CompuServe for the TIFORUM, a clearinghouse of TI information.
RICHARD ROSEEN: A trained engineer and electronics patent examiner with the U.S. Patent Office, Washington, D.C.
CHRIS BOBBITT: has experience with the mini and mainframe world, having

developed computer requirement specifications for a government agency, the U.S. Naval Research Lab.
GOALS: AEI will produce two operating systems, four computer languages and numerous utilities, business packages and applications for the Myarc 9640.

INTRODUCING ADOS:
AEI Disk Operating System

INTRODUCTION: ADOS is an advanced, modern, disk-oriented operating system that provides a comprehensive set of user services, resource management features, facilities and requests available to program operating under ADOS control.

DESCRIPTION: Like popular operating systems in the expensive microcomputing world such as MS-DOS and UNIX, ADOS supports a large, comprehensive library of utilities. These include an editor, assembler, file maagement utilities, a smart linker, a BASIC interpreter, file compression utilities and more. The smart linker is capable of resolving all references and declarations as well as supporting linked libraries of routines.
Unlike many operating systems, ADOS permits full "wildcards" at all prompts. A wildcard in operating system parlance is the ability to allow the operator to specify an incomplete or imprecise search specification of the item, such as a file, with a symbol being subsituted for the remaining characters. Other available capabilities will be such mini computer like features as mnemonic based command names, a complete set of file, disk and screen input output utilities available to the programmer as well as powerful debugging and diagnostic tools, along with access to all system functions. ADOS is the perfect operating system for the user, programmer and the systems designer.
ADOS frees the talented users of the 9900 world to enter the world of the "Value Added Reseller", marketing to profitable vertical markets as recognised as the stand-alone word processor users, or as exotic as any custom business or scientific application imagined. ADOS can be easily modified for a particular vertical application. User programs can interface with all operating system resources and utilites with an easy to understand, consistent protocol. This allows the development of user shells which are as easy to write as any functional program.
^A unique file management approach gives the ADOS, by design, a reliable environment, with file loss greatly reduced while disk transfers work faster.
COMPATIBILITY: The power of ADOS is not bought at the cost of compatibility. A complete library of both 99/4A and commercial Texas Instruments DNOS (a mature mini computer system) based applications are compatible. This suite of capabilities include utilities, business, entertainment and educational software. ADOS will DIRECTLY load binary files from machines that support these operating systems, thus opening a vast library of documented and tested software to the user at minimal expense and conversion effort. Many of these products could be marketed to established professional users used to paying many times the going rate for similar products.

INTRODUCING AMOS
AEI Multi-Tasking Operating System

AMOS is a general purpose multi-tasking, multi-user operating system for the Myarc 9640 computer. It is upwardly compatible with AEI Disk Operating System (ADOS) and provides an even more comprehensive suite of functions to programmers, developers and users running programs and packages under the control of AMOS.
DESCRIPTION: In addition to the wide variety of functions available under ADOS, AMOS provides a series of enhancements available only for thousands of dollars just months ago. While using the highly reliable ADOS file system, AMOS also supports a hierarchial file system with directories and sub directories,

along with file security and date stamping. Some of these features were unavailable outside of major mainframes and some minicomputers less than a year ago.

Sophisticaled applications are supported by AMOS. AMOS is versatile in that it shields programs from hardware imposed restraints yet allows for predictable expansion in hardware capacities in coming years. It is conceivable for a value added reseller to maintain a fully functional support system for a vertical market, responding to continuing demands using inexpensive hardware. This alleviates the problem of expanding support by making the reslers product highly profitable for the customers. The system impliments a byte addressable file space allowing individual files as large as 4 Gigabytes in length. AMOS provides a device independent interface for all system calls. AMOS therefor enables the developer to use new innovative hardware as it becomes available.

In addition to the sophisticated housekeeping ability enhanced by the AMOS multitasking capabilities, making a multi-user system is a simple configuration task using AMOS. AMOS is designed to be a networking operating system. Users can share data and programs, with AMOS resolving concurrent resource demands automatically. Sophisticated data security routines can be designed at the user level, based on levels of access with individual, protected directories. In these ways, AMOS is similar to professional operating systems for mini and mainframe systems.

Multiple Myarc 9640 computers in multi-node networks may contain an unlimited number of loosely coupled systems. AMOS can grow with any application, because the system manager can easily add services and utilities without modification of the system itself.

Like VMS on the DEC VAX computer, AMOS supports full background batch processing, even though these earlier systems sold thousands of units for tens of thousands of dollars. An example of background batch would be an internal software driven RAM disk and spooling utility, which in hardware add-ons still sells for hundreds of dollars in various configurations. Virtual memory constraints can be expanded through the use of external storage. It is truly a comprehensive operating system, so often described but seldom delivered.

INTRODUCING AC-BASIC and AC-PASCAL
by Acess Engineering Inc.

AC-BASIC: AEI Compiled BASIC is a TRUE compiled BASIC, contrary to the limited packages that have prolifereated in the MS-DOS world. AC-BASIC is a full, professional BASIC compiler that produces fast, optmised, 9900 assembly code. This bull featured BASIC is directly compatible with CBASIC on IBM and CP/M. It will, therefore, compile literally thousands of business, utility and application programs from those machines directly. A full compiler, untrammeled by schemes needed to optimise effort in large software houses is visibly more effective than some of the most widely used compilers in the industry today. Concentration of effort makes such focused products possible.

While compatible with CBASIC, AC-BASIC is not a clone, repeating well known problems with those versions. AC-BASIC is a complete enhancemebt if CBASIC, pushing software art further. AC-BASIC presents a professional alternative designed for large programming projects and professionally done smaller projects needing speed and optimization to make an attractive labor saving package for a selective ultimate user.

The tool kit included with AC-BASIC presents an attractive package in itself, in spite of the fact that AC-BASIC is the first full professional compiler for the 99/4A world. This modern compiler supports optional line numbers, multiple forms of commenting, reDIMensioning of arrays, common variable pools for chaining, and user defined

subroutines with global and local variables. AC-BASIC has a full range of flow commands and numeric functions. AC-BASIC is so advanced that it even produces threaded code, a capability worth learning about since it makes object files up to 35% smaller by removing redundant code and compression.

AC-PASCAL: AEI Compiled PASCAL is a remarkable derivation of Pacal; one which can easily prove claims for besting more than one of the most popular Pascals on the market for ANY machine. "Concurrent Pascal" was initially designed particularly to be used in designing multi-tasking programs with concurrent processes. As such, it is particularly well suited for designing powerful programs for use with the AEI Operating System (AMOS).
AC-Pascal is a full Pascal, closer in design and functionality to later fuller versions of Turbo Pascal (tm) than to UCSD Pascal. Conversion betwen these multiple forms of Pascal is relatively simple. As a result, vast amounts of program code is available. Like AC-BASIC, this compiler produces 100% 9900 assembly code that is optimised and threaded. AC-Pascal is particularly suited for designing system software. In fact, both AEI packages "ADOS" and "AMOS" were written with AC-PASCAL.

AVAILABILITY:

Both AC-BASIC and AC-PASCAL are available exclusivley for the Myarc 9640 (Geneve) computer with at least 640k o' RAM and a single disk drive. Both packages function equally well under ADOS and AMOS, both by AEI. AC-BASIC and AC-PASCAL will be available by 1 May, 1987.

For futher information, contact AEI, 9120 Rhode Island Ave., College Park, Maryland 20740

Converting the TI-WRITER/Word Processor to run within memory

By following the step by step instructions in this article you will be able to convert your TI-WRITER cartridge and disk to run within memory using the Mechatronic Gram Card. The cartridge will perform the same except for the fact that loading either the Editor or Formatter files will be instantaneous since they will now be stored in 5 of the Gram/Ram segments in the card.
Installing the new TI-WRITER

1) Initialize a new disk.
2) Copy the files CHARA1, EDITA1, EDITA2, FORMA1, and FORMA2 onto this disk in the order they are listed here: CHARA1, EDITA1, EDITA2, FORMA1, FORMA2. 3) Next use a sector editor program such as DISK-PATCH or DISK FIXER to modify the sectors >20, >28, >48, >4D, and >6D. Only the first 6 bytes of each sector need to be modified. Below are listings of the original 6 bytes of the sector and the values to which they should be changed:

NOTE: If the bytes you see in the sector are not the same as the originals listed here DO NOT continue. Repeat steps 1 and 2. Either you are not working with a newly initialized disk or you have copied the files in incorrect order!

SECTOR >20

Original:

0000 0800 07FA

Change to:

A5A5 C000 0800

SECTOR >28

Original:

FFFF 2000 2000

Change to:

A5A5 8000 2000

SECTOR >48

Original:

0000 0410 A000

Change to:

A5A5 A000 0410

SECTOR >4D

Original:

FFFF 2000 A000

Change to:

A5A5 E000 2000

SECTOR >6D

Original:

0000 0D08 BFF8

Change to:

5A5A 6000 2000

4) Insert your TI-WRITER cartridge in the console and select 1 for TI BASIC. Bring up the Gram Card screen by entering CALL GRAM. When the menu shows up select option 5 for SAVE GROM. Enter 6000 for the start address and 7800 for the end address. Press enter to the GROM-ADR prompt to default to the >9800 Grom address. Save the Grom in the file name DSK1.TI/WG3. After you receive the message COMMAND COMPLETED press function = <quit>, to return to the title screen.

5) Remove the TI-WRITER cartridge and re-boot the system. Select 1 for TI BASIC. Enter CALL GRAM to get to the Gram Card menu and select option 1 LOAD (G)RAM WITH PROGRAM. Select a base of >9800 by pressing enter to the GRAM-A: prompt, and enter the file name DSK1.TI/WG3. After the

Notes:

1) This file was written using the new TI-WRITER installed in a Mechatronic Gram Card.

2) The FORMATTER will function only in the first module library of the Gram Card. <Base >9800) I believe this is due to the FORMATTER loading data from the GROM in the cartridge.

program returns the message COMMAND COMPLETED return using FCTN = (QUIT).

6) Again select 1 for TI BASIC. This time enter CALL EDITMEM. This will bring you to the memory editor of the Gram Card. Press G for grom and enter the address 7700 at the START MEMORY: prompt. Press enter to the GRAM-ADDRESS: prompt for an address of >9800. You will now be able to change the gram memory of the TI-WRITER cartridge. In the area from >7700 to >7758 there is nothing but zero's. Change these to the values listed below.

```
7700   31 08 00 A8 00 C0 00 31
7708   20 00 8F 9D 00 80 00 31
7710   04 10 8F 1D 00 A0 00 31
7718   20 00 AF 10 06 80 00 31
7720   04 10 AF 30 06 A0 00 BF
7728   00 20 00 00 00 00 00 00
7730   00 00 00 00 00 00 00 00
7738   00 00 00 00 00 00 00 00
7740   31 20 00 8F 1D 00 E0 00
7748   35 0D 08 8F 3C F8 8F DD
7750   00 BF 00 B7 04 00 00 00
```

After modifying the gram press function 9 <back>, to return to the EDITMEM screen. There are ten other bytes you must modify in the Gram. Again press G for Grom. Enter 62F8 for the address and press enter to the GRAM-ADDRESS: prompt. Change the memory listed below to the new values.

Original:

```
62F8  06 00 36 05 62 E4 06 67
6300  77 06 66 63 01 66 86 06
```

Change to:

```
62F8  06 00 36 05 62 E4 06 77
6300  00 05 63 61 01 66 86 06
```

After modifying the gram press function 9 <back>, to return to the EDITMEM screen. Once more press G for Grom. Enter 6320 for the address and press enter to the GRAM-ADDRESS: prompt. Change the memory listed below to the new values.

Original:

```
6320  06 67 77 06 66 63 04 66
```

Change to:

```
6320  06 77 40 05 63 61 04 66
```

After modifying the Gram data press function 9 <back> to return to the Edit Memory main screen and then press function 5 <begin> to return to TI BASIC.

7) Now that you are back in BASIC enter CALL GRAM to get to the Gram Card main menu. You must now save the modified Gram as you did before in step 4. Select option 5 to save the GROM and again save addresses 6000 to 7800 and the file name DSK1.TI/WG3. Exit back to the main screen.

8) The last step is to set up a load file for the new TI-WRITER. Use a text editor such as TI/WRITER or EDITOR/ASSEMBLER. If you use TI/WRITER make sure you are using "fixed" mode so there are no control codes embedded in the file. If you see the little CR characters in the file then you are in "word-wrap" mode and this is wrong! Type in the lines:

```
DSK1.TI/WG3
DSK1.CHARA1
DSK1.EDITA1
DSK1.EDITA2
DSK1.FORMA1
DSK1.FORMA2
```

Make sure you have a blank line as the last line in the file! Save this file to disk as DSK1.WR. You are now done! To load the new TI-WRITER just reset the system (including the P-BOX to wipe out the current Grams) and select option 2 GRAM CARD >9800 on the menu. Then from the Loader screen press 6 DSK1.WR and you now have your new program running! No more waiting for the EDITOR or FORMATTER files to load! This should save a lot of time when you are working on text files.

J & K H Software

4911 South 31st Street
Arlington, Virginia 22206-1655
(703) 820-4131

PRICE REDUCTIONS

Video Titles I™

Video Titles I is a program written in BASIC for the TI-99/4(A) which provides a means of producing custom titles for video recordings without the aid of a camera. Features of the program include: three proportionally spaced character sets (a maximum of any two may be used for each title), automatic centering of each title line, variable spacing with automatic eye correction, 26 foreground/background color combinations and multiple screen division with scrolling. **Was $29.95; Now $19.95** [VT1/disk] or **$22.95** [VT1/tape]

Video Titles II™

Video Titles II is a program written in Extended BASIC for the TI-99/4(A) which provides a means of producing automated sequences of custom titles for in-store advertising or video recordings. Features of the program include: three proportionally spaced character sets, choice of left, centered or right justification for each title line, variable spacing with automatic eye correction, choice of four frame styles for each title, animation/overlay of custom designs such as logos, etc., and storage for 40 titles, 40 sprite patterns and 10 title sequences. A collection of sub-programs is linked via menus which guide the user throughout the entire process. **Was $49.95; Now $34.95** [VT2/disk]

Video Titles II Accelerator™

The Video Titles II Accelerator is a companion program to Video Titles II. Written in assembly language, it loads the entire Video Titles II Data Base into Memory Expansion—then allows title sequences to be displayed at lightening fast speed. Adds many fine tuning controls to existing Video Titles II capabilities plus new ones including linking sequences. **Was $24.95; Now $19.95** [VT2A/disk]

Video Titles III™

Video Titles III is a group of three programs written in BASIC for the TI-99/4(A) which provide a means of easily producing custom titles for your own programs. Each of the three programs produces a different style of proportionally spaced lettering (bold, computer or miniature). Other features include automatic justification (left, center or right) and variable spacing with automatic eye correction. A fourth program is included which shows one of the many possible ways in which the output of all three Video Titles III programs can be used to create a custom title for your own program. **Was $19.95; Now $12.95** [VT3/disk] or **$14.95** [VT3/tape]

Video Titles Combination™

All of the Video Titles products listed above (VT1, VT2, VT2A & VT3) on two disks for one low price. **Was $99.95; Now $54.95** [VT-COMBO/disk]

Super Extended Basic™

Super Extended Basic is a powerful extension to the TI Extended Basic programming language. SXB is comprised of over one hundred TMS9900 Assembly Language subroutines which substantially expand your programming capabilities on the TI Home Computer. The subroutines are easily invoked with the Extended Basic LINK subprogram. In essence, it is now possible to access the raw power of assembly language with only a knowledge of Extended Basic and the information provided in the SXB reference manual. All SXB subroutines are invoked with the following format: CALL LINK("subroutine"[,arguments]).

Included with the basic package are the first six issues of the SXBrief Newsletter, a monthly publication dedicated solely to SXB and related items. Most issues include an additional assembly language USRSUB (user subroutine) which can be added to SXB at execution time. Each newsletter fits nicely into the custom three-ring SXB reference manual binder.

". . . takes the TI-99/4A to a level which far exceeds the wildest imagination of a Basic programmer." —The New York Times PCUG

"Performance = A; Ease of Use = A; Documentation = B; Value = A; This is one of the most valuable programming tools for the Extended Basic programmer. Granted, the programmer gets few of the enhanced graphics routines found in other packages, but in their place he gets the routines that do more *meat and potatoes* chores, such as string handling and sorting. Like Extended Basic itself, the SXB package is not specific to one task. It is a *complete* enhancement package in that it makes every programming task simpler, not just those involving graphics or data handling alone. It would be great if many of these routines could have been included in the Extended Basic cartridge itself." —MICROpendium (February, 1985) **Was $99.95; Now $49.95** [SXB/disk]

S.X.B.™ Assembly Listing

Learn the secrets of Super Extended Basic. Now available—the original assembly language listing of S.X.B. (almost 2,700 lines of coding). *This item may only be ordered with S.X.B. or following our receipt of the S.X.B. software registration card.* **Was $39.95; Now $34.95** [SXB/list]

Multi-Disk Informer™

•(Version Requiring SXB)

The Multi-Disk Informer is a program written in Extended BASIC with SXB subroutine calls which inputs all your disk directories and produces two master listings: 1) the FILE REPORT—a sorted list of filenames identifying type, protection, size and disk ID (name and input sequence); and 2) the DISK REPORT—a sorted list of disknames identifying input sequence, up to eight files resident on each disk, sectors used, free and total, and counts by file type. **Was $19.95; Now $12.95** [MDI-SXB/disk]

•(Stand Alone Version)

Same as MDI-SXB except that SXB is not required. **Was $29.95; Now $17.95** [MDI-SA/disk]

• • • • ORDER BLANK • • • •

Qty	Item No. & Price	Price
___	VT1/disk @ $19.95	_____
___	VT1/tape @ $22.95	_____
___	VT2/disk @ $34.95	_____
___	VT2A/disk @ $19.95	_____
___	VT3/disk @ $12.95	_____
___	VT3/tape @ $14.95	_____
___	VT-COMBO/disk ...	
	... @ $54.95	_____
___	SXB/disk @ $49.95	_____
___	SXBrief Newsletter ...	
	... #7-18 @ $10.00	_____
___	SXBrief Newsletter ...	
	... #19-30 @ $10.00	_____
___	SXB/list @ $34.95	_____
___	MDI-SXB/disk ...	
	... @ $12.95	_____
___	MDI-SA/disk ...	
	... @ $17.95	_____
	Sub-Total	_____
	4.5% Tax (VA only)	_____
	Regular S&H $4.00	_____
	Air Mail S&H $10.00	_____
	Total (U.S. funds)	_____

Payment: []Check #_____ []Visa []MC
Card #_____ _____ _____ _____
Expiration Date: _____
Signature: _____
Name: _____
Phone: _____
Address: _____
_____
_____
_____

INTRODUCTION TO "c99"
by Ronald G. Albright, Jr.

First, let me clarify my qualifications. I have absolutely no qualifications to discuss this language or the "c99" compiler specifically, other than a desire to learn more about this language. For the first time, Clint Pulley has made it possible for users of the TI 99/4A Home Computer to use a version of the language, "C", on our machine. It is an implimentation based on "small-c", a public domain version of the "C" language which was published as a source code listing in the May, 1980 issue of Dr. Dobb's Journal (Number 45). Like "small-c", "c99" is a slightly restricted but otherwise syntactically identical to true "C". It allows the use to be exposed to the language that, according to PC Magazine, has more versions (at least a dozen C compilers) of it than any other language. It is the language that most application software packages are written in. It is used by Visicorp, Microsoft and Digital Research in major software projects.

So what does the "c99" package do"? Well, Clint Pulley gives you a basic set of documentation files on disk with the package. The documentation does not teach you C, but does show you how to run the compiler and what the compiler does and does not support. You will have to get a book to learn C, which is what 99/4A users had to do to learn Forth anyway. (See list at the end of this article.) Then you have to write a C source code listing. A couple are provided by Pulley with the c99 package. One, a c99 version of the classic prime number sieve benchmark which appeared in BYTE Magazine. The c99 source code is shown below as Listing 1. While I know nothing about C, a book has told me that comments are enclosed with /* and */ to show the beginning and end of comment areas; which can extend over several lines. The main routine that a program will perform starts with the word "main". This is a special funtion that must be found in any 'C' program. Any values or arguments that are being given to the program at the outset are enclosed in parentheses. The beginning and end of a group of statements are marked with braces; a C program will always end with a closed brace. Like Pascal, each complete program statement ends with a semicolon.

Listing 1

```
/* A c version of the Sieve program
as suggested by KNUTH.
*/
#define true 1
#define false Ø
#define size 819Ø
#define sized 8191
/**/
int flags[sized];
#asm
 AORG >833Ø
#endasm
int i,k,prime,count,iter,strikeout;
#asm
 RORG
#endasm
main()
{ puts("1Ø iterations\n\n");
  iter=1;
  while(iter++<=1Ø)
  { strikeout=true;
    count=Ø;
    i=Ø;
    while(i<=size) flags[i++]=true;
    i=1;
    while(i<=size)
    { if(flags[i])
      { prime=i+i+1;
        ++count;
        if(strikeout)
        { if((k=((prime*prime)-1)>>1)<size)
          while(k<=size) {flags[k]=false; }
          else                          k=k+prime;}
          { strikeout=false;
            continue;
          }
        }
      }
      ++i;
    }
    puts(" working...\n");
  }
  puts("\nDone!\n");
}
```

Once the c99 source code has been

Listing #2

```
*c99 v1.3
 REF C$INIT,C$CIND,C$DIV,C$REM,C$ASR,C$ASL,C$EQ,C$NE
 REF C$LT,C$LE,C$GT,C$GE,C$ULT,C$ULE,C$UGT,C$UGE
 REF C$LNEG,GETCHA,GETS,PUTCHA,PUTS,LOCATE,POLL,EXIT
 DEF MAIN,START
START B @C$INIT
FLAGS BSS 16382
 EVEN
 AORG >8330
I BSS 2
 EVEN
K BSS 2
 EVEN
PRIME BSS 2
 EVEN
COUNT BSS 2
 EVEN
ITER BSS 2
 EVEN
STRIKO BSS 2
 EVEN
 RORG
MAIN
 LI 8,C#1+0
 BL 15
 BL *12
 DATA PUTS
 INCT 14
 LI 8,1
 MOV 8,@ITER
C#2
 MOV @ITER,8
 INC 8
 MOV 8,@ITER
 DEC 8
 BL 15
 LI 8,10
 MOV *14+,9
 BL @C$LE
 ABS 8
 JNE $+6
 B @C#3
 LI 8,1
 MOV 8,@STRIKO
 CLR 8
 MOV 8,@COUNT
 CLR 8
 MOV 8,@I
C#4
 MOV @I,8
 BL 15
 LI 8,8190
 MOV *14+,9
 BL @C$LE
 ABS 8
 JNE $+6
 B @C#5
 LI 8,FLAGS
 BL 15
```

(continued)

```
 MOV @I,8
 INC 8
 MOV 8,@I
 DEC 8
 A 8,8
 MOV *14+,9
 A 9,8
 BL 15
 LI 8,1
 MOV *14+,9
 MOV 8,*9
 B @C#4
C#5
 LI 8,1
 MOV 8,@I
C#6
 MOV @I,8
 BL 15
 LI 8,8190
 MOV *14+,9
 BL @C$LE
 ABS 8
 JNE $+6
 B @C#7
 LI 8,FLAGS
 BL 15
 MOV @I,8
 A 8,8
 MOV *14+,9
 A 9,8
 MOV *8,8
 ABS 8
 JNE $+6
 B @C#8
 MOV @I,8
 BL 15
 MOV @I,8
 MOV *14+,9
 A 9,8
 BL 15
 LI 8,1
 MOV *14+,9
 A 9,8
 MOV 8,@PRIME
 MOV @COUNT,8
 INC 8
 MOV 8,@COUNT
 MOV @STRIKO,8
 ABS 8
 JNE $+6
 B @C#9
 MOV @PRIME,8
```

(continued)

```
 BL 15
 MOV @PRIME,8
 MOV *14+,9
 MOV 8,7
 MPY 9,7
 BL 15
 LI 8,1
 MOV *14+,9
 S 8,9
 MOV 9,8
 BL 15
 LI 8,1
 MOV *14+,9
 BL @C$ASR
 MOV 8,@K
 BL 15
 LI 8,8190
 MOV *14+,9
 BL @C$LT
 ABS 8
 JNE $+6
 B @C#10
C#11
 MOV @K,8
 BL 15
 LI 8,8190
 MOV *14+,9
 BL @C$LE
 ABS 8
 JNE $+6
 B @C#12
 LI 8,FLAGS
 BL 15
 MOV @K,8
 A 8,8
 MOV *14+,9
 A 9,8
 BL 15
 CLR 8
 MOV *14+,9
 MOV 8,*9
 MOV @K,8
 BL 15
 MOV @PRIME,8
 MOV *14+,9
 A 9,8
 MOV 8,@K
 B @C#11
C#12
 B @C#13
C#10
 CLR 8
```

written, you must then run the c99 compiler. It is loaded with the Option 5 of the Editor/Assembler package. The program, once loaded prompts you for the source code file name, the output file name and then takes off. It is very fast. The end product to your output file is a TI 99/4A assembly language source code in D/V80 format. This is suitable and, indeed, ready for assembly into runnable object code by the E/A assembler. The outputted assembly language source code for the above C routine is shown in Listing 2. It will certainly look inefficient and, yes, even weird to experienced assembly language programmers. But it assembled fine for me.

Listing 2

CONTINUED

```
 MOV 8,@STRIKO
 B @C#6
C#13
C#9
C#8
 MOV @I,8
 INC 8
 MOV 8,@I
 B @C#6
C#7
 LI 8,C#1+16
 BL 15
 BL *12
 DATA PUTS
 INCT 14
 B @C#2
C#3
 LI 8,C#1+29
 BL 15
 BL *12
 DATA PUTS
 INCT 14
 B *13
C#1 BYTE 49,48,32,105,116,101,114,97,116,105,111,110
 BYTE 115,10,10,0,32,119,111,114,107,105,110,103
 BYTE 46,46,46,10,0,10,68,111,110,101,33,10
 BYTE 0
*errors=0
 END
```

Once assembled, you are ready to run your program, now in D/F 80 file format. You load it with Option 3 of the E/A cartridge, then load the c99 support routines the same way, hit ENTER, enter the program name "START", and the program executes as fast as classically written assembly language code. It is fun and efficient. The best part is: it is Fairware. It is available from the author to try [also from Ryte Data version 3.0] and if you use it, you then pay for it. A $20.00 payment is suggested. Pulley vows continued support if the users support him. New improved versions are planned. Pulley's address is:

Clint Pulley
38 Townsend Avenue
Burlington, Ontario L7T 1Y6

BIBLIOGRAPHY

THE C PROGRAMMING LANGUAGE, Brian Kernighan and Dennis Ritchie, Englewood Cliffs, N.J.; Prentice-Hall, 1978. [Not for beginners but considered to be the "bible" for this language; written by the originator of the language (Ritchie)].

THE C PUZZLE BOOK, Alan Feuer; Englewood Cliffs, N.J., Prentice-Hall, 1982.

THE C PRIMER, Les Hancock and Morris Krieger; New York, McGraw-Hill, 1982

LEARNING TO PROGRAM IN C, Thomas Plum, Cardiff, N.J., Plum Hall Inc., 1983.

C PROGRAMMING GUIDE, Jack Purdum, Indianapolis, Que Corp., 1983.

8K DSR RAM PROJECT

John Clulow (GEnie - J.CLULOW)

Part One of a Three Part Series

The present article describes the addition of 8K of battery backed RAM to the Peripheral Expansion Box. This RAM is decoded in the address space >4000 - >5FFF and is switch selectable in the CRU base range >1000 - >1700 or >1800 - >1F00. It requires 8 - 10 hours to complete. A parts kit is available from Bud Mills Services, 166 Dartmouth Drive, Toledo, Ohio 43614. The kit is expected to cost about $40. Component values have been selected to be consistent with parts Bud has on hand. However, if you choose to buy your components elsewhere, other values will work equally well.

The second part of the project implements one of John Johnson's ideas; addition of 8K of battery backed RAM in the address space >4000 - >5FFF such that this RAM is enabled only when a DSR ROM or RAM is NOT enabled. It is called non-DSR RAM. In effect, this 8K extends the size of the low memory to 16K from >2000 to >5FFF. This modification to the 8K DSR card requires only takes a few additional parts. Two additional IC's are required; a 74LS259 and a MM6264LP-15. No new sockets are needed because both IC's are piggy-backed. This project also uses ten 1N914 diodes three 2.7K resistors, one 10K resistor, and a 2N2222 transistor.

The third and final part of the project will add a real time clock/calendar to the 8K card.

If you should attempt this project and run into problems, please feel free to contact me. Anyone who cannot access GEnie may obtain the loader, menu and clock software and docs for $3 by writing to me. However, please do not request this until you have actually completed the project. John Clulow, 345 W. South Boundary, Perrysburg, OH 43551, (419) 874-8838.

BACKGROUND

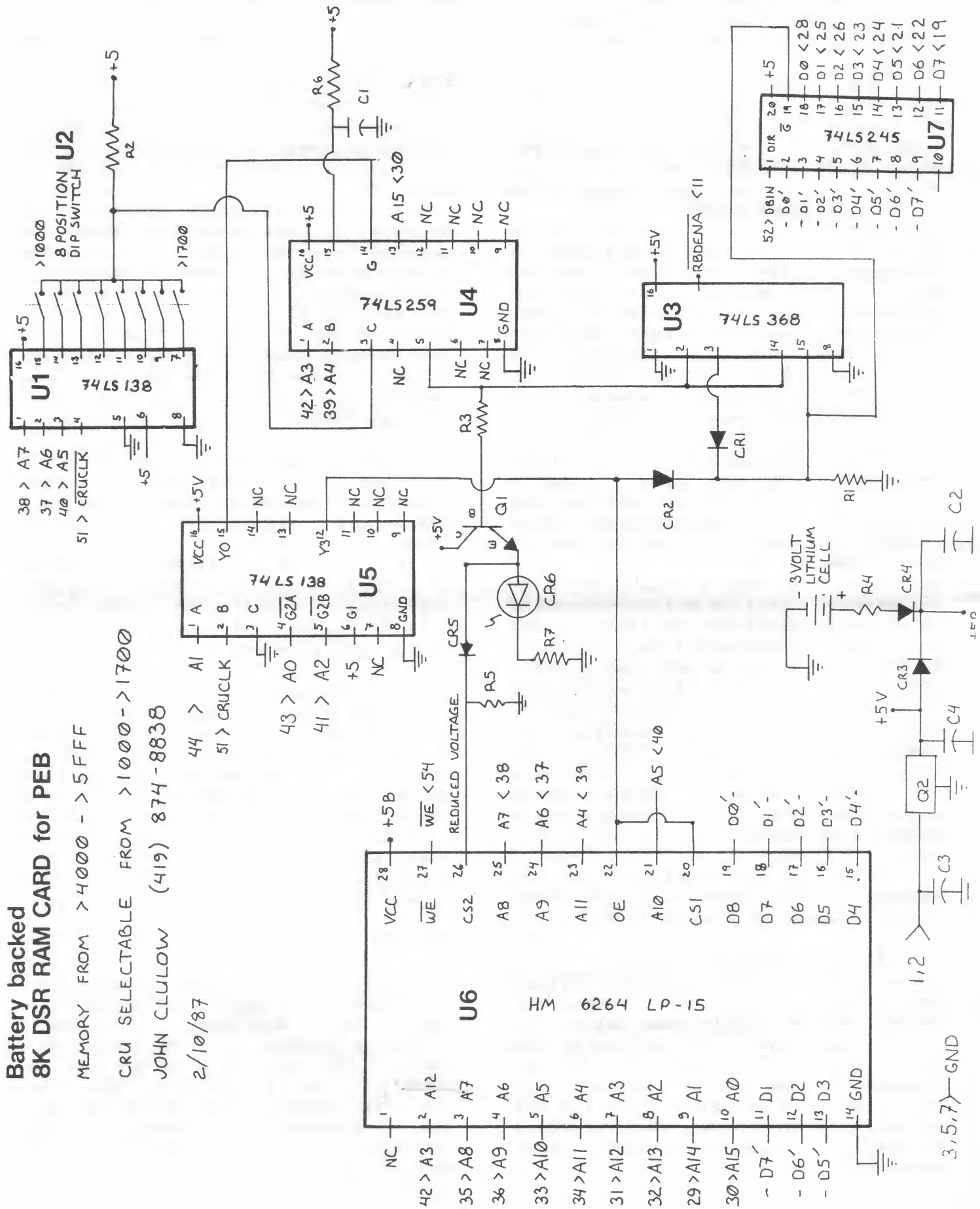
The 8K DSR RAM project was undertaken to make a unique piece of software available to more users: the John Johnson MENU program for the Horizon Ramdisk. I own three Horizon Ramdisks and use John's program routinely. On power-up it provides a menu of virtually all significant pieces of TI software. These can be selected in seconds with a single key press. No module need be in the GROM slot in order to use the power-up MENU program. Although I believe use of this program would justify the purchase of Horizon Ramdisk kit (~$165 US), I wanted to make the menu idea available to those who, for one reason or another, will not have a Horizon Ramdisk.

RAM in the DSR address space is accessed by polling routines. Assembly language programs using this RAM may therefore incorporate power-up, interrupt service, and device service routines. Software developed to utilize these linkages, including a version of John's MENU, will be placed in the program library on the GEnie network.

Subsequently, John Johnson indicated that it was possible to have 8K RAM in the >4000 - >5FFF address space whenever a DSR ROM or RAM was not selected. John's idea worked and will be described in the second part of this series. This RAM is also battery backed and can be used like ordinary CPU RAM with the exception that it cannot directly access DSR ROM or RAM.

WARNING

THIS PROJECT SHOULD ONLY BE ATTEMPTED BY PERSONS WITH SUFFICIENT EXPERIENCE IN THE REQUIRED TECHNIQUES. EXTENSIVE DAMAGE TO THE PEB POWER SUPPLY AND ANY CARDS IN THE PEB CAN OCCUR IF SHORT CIRCUITS ARE NOT CORRECTED BEFORE POWERING UP THE PEB. THE SCHEMATIC DIAGRAM AND INSTRUCTIONS FOR THIS PROJECT ASSUME PRIOR FAMILIARITY WITH ELECTRONICS PRINCIPLES INVOLVED. IF YOU ARE NOT


Battery backed
8K DSR RAM CARD for PEB
MEMORY FROM >4000 - >5FFF
CRU SELECTABLE FROM >1000->1700
JOHN CLULOW (419) 874-8838
2/10/87
U1
74LS138
8 POSITION DIP SWITCH
U2
>1000
>1700
U3
74LS368
U4
74LS259
U5
74LS138
U6
HM 6264 LP-15
U7
74LS245
3 VOLT LITHIUM CELL
REDUCED VOLTAGE
GND
+5
+5V

of the board. Then DOUBLE CHECK ALL CONNECTIONS using a continuity tester. Finally use short pieces of wire-wrap wire to tie together the bundles of wires going to the connector.

TESTING THE CARD

Insert all IC's except U7, the 6264LP-15. Insert the 8 pin DIP switch and turn on switch 1. Use the continuity tester to insure that only this connection is closed. Do not insert the lithium cell. Solder a 12" wire to pin 26 of the 6264LP-15 socket so that the voltage on this pin may be tested when the CRU line is activated.

Remove all cards from the PEB except the interface card. Plug the 8K DSR card into the PEB. Power-up the PEB and then the console. The LED should not light and you should get the normal TI Master Title Screen. If this is not the case, power-down immediately and determine the source of the problem.

Using Mini Memory EASY BUG or the E/A DEBUG, activate CRU >1000 (or >1800) with the C command. The LED should light (red). Check the voltage on pin 26. It should be about 2.5 volts. If either condition is not met, retrace all connections to identify the problem.

Power-down and remove the card from the PEB. Remove the test wire attached to pin 26 of the 6264LP-15 socket. Insert the lithium cell. Check the voltage from the top of the cell (+) to ground. It should be about 3 volts. Insert the 6264LP-15 IC. Test the voltage on pin 28 and re-test the voltage on the battery. If either the battery voltage or the voltage on pin 26 drops significantly relative the the test without the IC inserted, the IC is NOT Low Power and cannot be used. (Occasionally 6264LP-15 chips from Malaysia are not low power even though they are labeled LP!)

Again use the C command of DEBUG or EASY BUG to activate CRU >1000 (or >1800) and try writing to memory in the range >4000 - >5FFF with the M command. Test the retention of memory by powering down the PEB after data has been written to the memory and while the LED is still lit. No data should be lost or altered.

To test all memory with DEBUG, select the CRU to turn on the card with .C 1000,1 <enter> and then entering a 1 for the CRU value. Then place the test value in >4000 with the M command. This value can be written to all memory locations with .N 4000 40001 1FFF and these locations can be subsequently tested with .P 4000 5000 1000. After the card passes all tests successfully, it should be tried with the disk controller in place.

------ PARTS LIST ----------

RESISTORS - 1/4 Watt

R1 R2 R5 - 2.2K ohm
R3 R6 - 10K ohm
R4 - 360 ohm
R7 - 220 ohm

CAPACITORS - Tantalum

C1 - 2.2 MFD
C2 - 22 MFD
C3 - 10 MFD
C4 - 0.1 MFD

DIODES

CR1 CR2 CR5 - 1N914 (silicon) } Test all diodes
CR3 CR4 - 1N34 (germanium) }
CR6 - Standard LED

INTEGRATED CIRCUITS

U1 U5 - 74LS138
U3 - 74LS368
U4 - 74LS259
U6 - HM6264LP-15

MISCELLANEOUS

Q1 - Transistor 2N2222
Q2 - Regulator 7805
U2 - 8 Position DIP Switch
U7 - 74LS245

SURE ABOUT ANY PART OF THIS PROJECT, MAKE SURE YOU OBTAIN EXPERT ADVICE BEFORE PROCEEDING.

FABRICATION OF THE BOARD

A 60 pin - 1/10 inch spacing plugboard is required for this project. It is imperative that the board be constructed such that it is impossible to plug in backwards: It must extend from the Attleboro connector to the slot in the front of the PEB.

An adequate board can be constructed from a Radio Shack Jumbo Component Perfboard (276-147). This board does not have a connector. If a scrap board can be located with more than 60 pins - 1/10 inch spacing, an appropriate connector can by sawed off and fixed to the perfboard. If a scrap board cannot be found, an acceptable connector can be obtained from DIGI-KEY; part C8-30. The connector piece (about 3 X 1") should be cemented to the perfboard with five minute two-part epoxy. (Sand and apply well mixed cement to both surfaces before clamping, and allow 60 minutes cure before working with the board.) It should be positioned such that the front edge of the perfboard fits into the slot in the front of the PEB with the component side toward the disk drives. A 1 X 1/2" notch should be cut in the bottom front edge of the perfboard, extending upward, so the the board will slide all the way into the box.

BOARD PREPARATION

Use fine steel wool to remove the tarnish from the copper solder pads on the perfboard. Position the sockets as shown on the layout diagram and solder all pins to the solder pads. Bend the adjacent pins of U1 and U2 so that they touch and solder them together. Note that U2 is moved down one hole from U1.

Place tape over the bottom 1/4 inch of both sides of the edge connector pins. Label the tape with pin numbers. Pins start at the back of the PEB with even numbers on the component side (facing the disk drives) and odd numbers on the solder side (facing the PEB power supply.) Apply solder to the top of each connector pin. With the board prepared in this way, wires can be attached to pins by heating the pin -- no solder need be added.

Solder a heavy, bare wire along the front edge of the board above the slot to act as a ground bus.

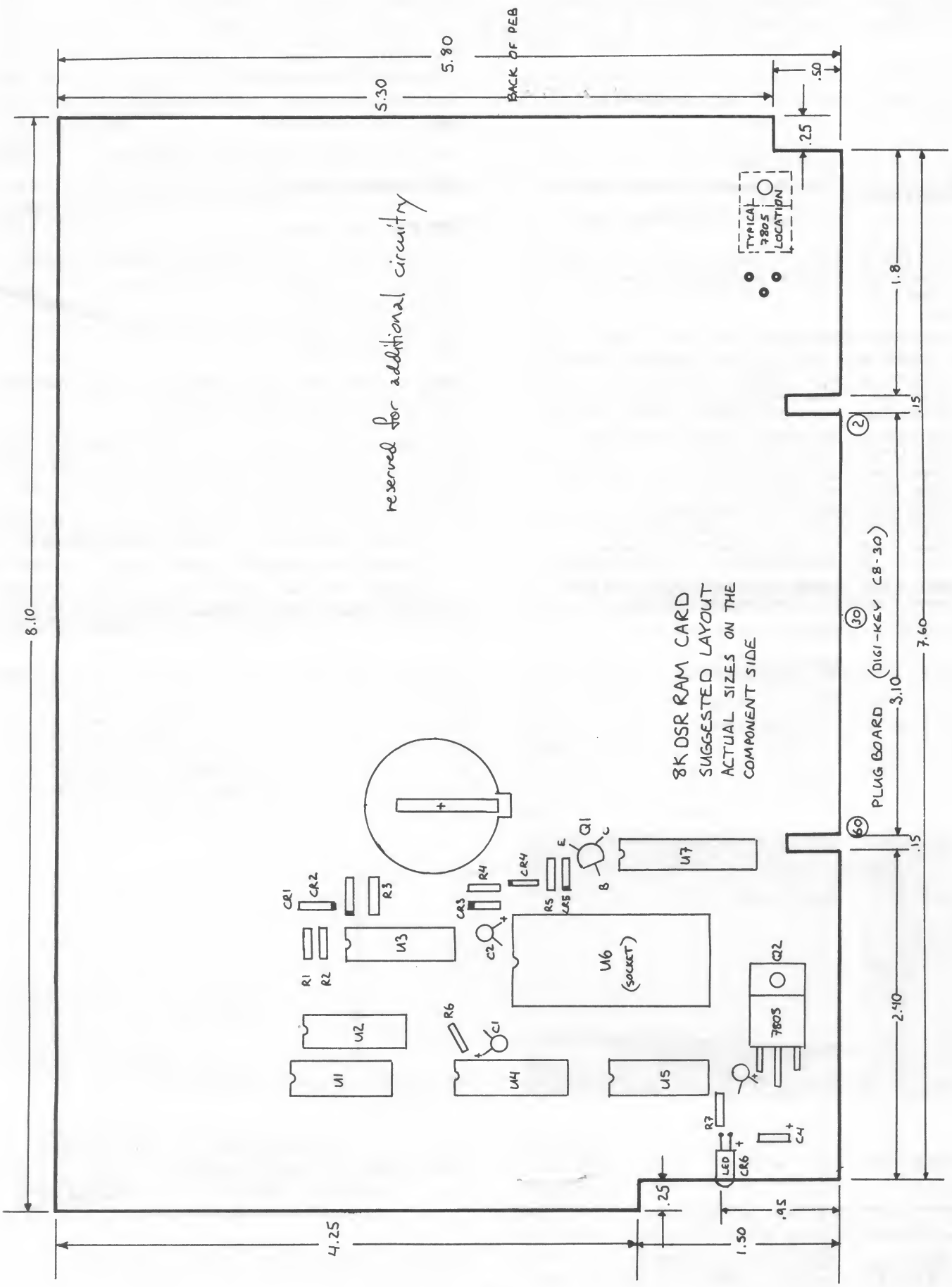
CONSTRUCTION SUGGESTIONS

Use wire wrap wire (e.g., Radio Shack 278-501) to make all connections. Use an appropriate stripping tool (such as the one in Radio Shack 276-1570). It is a good idea to use two colors of wire; one for connections between components on the board and another for all connections to the edgecard connector. This will help to facilitate checking and identification of errors.

Use a highlighting pen to mark off connections on the circuit diagram as each is made. Position the LED so that when the board is in the PEB, the LED is lined up with the front window. Use the RED and COMMON leads of the bi-color LED for this project. The COMMON (longest) lead goes through the resistor R6 to ground. The GREEN (shortest) lead will be used in the non-DSR RAM project.

To use CRU lines from >1000 - >1700 (recommended), connect R2 to pin 5 of U5 as shown. To use CRU lines >1800 - >1F00 instead, connect R2 to pin 7 of U5. In this case, the top switch will be >1800 and the bottom switch >1F00.

After all connections have been made, use a continuity tester to make sure there is not a short from ground (edgecard pins 3,5,7) to both edgecard +5 (pins 1 & 2) and the output side of the regulator -- the regulator pin closest to the bottom


8K DSR RAM CARD
SUGGESTED LAYOUT
ACTUAL SIZES ON THE
COMPONENT SIDE
reserved for additional circuitry
BACK OF PCB
5.80
5.30
.50
.25
TYPICAL 7805 LOCATION
1.8
.15
2
30
60
PLUG BOARD (DIGI-KEY CB-30)
3.10
7.60
2.40
8.10
4.25
1.50
.95
.25
U1
U2
U3
U4
U5
U6 (SOCKET)
U7
7805
Q1
Q2
C1
C2
C4
R1
R2
R3
R4
R5
R6
R7
CR1
CR2
CR3
CR4
CR5
LED
CR6

SUBSCRIPTION COST:

$14.00 (US) for 12 issues $20.00 Cdn. all via First Class Mail.
$19.00 (US) overseas Airmail Delivery.

BACK ISSUES:

Back issues are available for $2.00 each, **subscribers** only for these issues —

V 2 sold out.
V 3 TMS 9995 Memory Map & specs numeric keypad project.
V 4 XB II **plus** by Mechatronic, Myarc 128k card, Autofire joystick project.
V 5 32k internal memory project, DS/DD Ramdisk, Auto power-up project.
V 6 Myarc 256k computer, "C" compiler, RAM/GRAM card.
V 7 Maximem review, EEROM programmer, Sense and control card.
V 8 Sold out.
V 9 Dual disk controller project, RESET switch project.
V 10-11 Double Issue: LA TI FAIR, Bill Gronos on Assembly, Super load switch, EPROM programmer.
V12.5 Myarc 640k Geneve, Console speed upgrade, Video monitor filter, Gramkracker review, ATRONIC products.
V14.5 Expansion Box 8k 'module,' Super Clock Support, Basic Compiler, 9938 Video Chip.
V15 Video upgrade @ 15mHz, Quad Density TI Disk Controller, 80 Column Display preview.
V16 P Box Modification, Multi-Module project.
V17 Tigercub, Eprommer update, Quad density update.
V18 Power Supply, RAVE 99 Keyboard, Year Review.
V19 Geneve, c993.0, RAM Disk backup, Hi-Speed cassette.
V20 99AT, T.O.D., Horizon modification, Rapid Copy 32k project, GRAM card software, etc.

R/D COMPUTING is published monthly by Ryte Data in Haliburton, Ontario.

All material is from sources believed to be accurate. The publisher takes no responsibility for errors, omissions or misprints. Articles may be reprinted with credits giving source and address by users groups for publication.

Articles dealing with Texas Instruments 99/4A and 9900 based computers are published. Special attention is given to projects and information on upgrading and modifying the 99/4A console and system.

Please send any information on products, hardware, software, or modifications to our address. We cannot accept responsibility for materials submitted and, unless stated otherwise, will assign all manuscripts etc. for publication.

Only manuscripts with sufficient postage and self-addressed mailer will be returned.

Un-classified ad rates are $1.25 per 40 character line. Count ALL characters, spaces, etc. and submit with payment to our address. Ads are run in order received. Deadline is 21st of each month.

YOU MUST NOTIFY US OF YOUR CHANGE OF ADDRESS SIX WEEKS IN ADVANCE. PLEASE SEND A CHANGE OF ADDRESS CARD WITH YOUR OLD AND NEW ADDRESS.

Ryte Data
210 MOUNTAIN STREET
HALIBURTON, ONTARIO K0M 1S0
705 457-2774